(2)

Connectionist Network Supercomputer Project

A Collaboration of the
University of California
and the
International Computer Science Institute

John Wawrzynek

March 1994

Personnel

- *Faculty*
  Jerry Feldman
  Nelson Morgan
  John Wawrzynek
- *Staff*
  James Beck
  Phil Kohn
  David Johnson
  Bertrand Irissou
- *Visiting Researchers*

  Silvia Mueller
  Arno Formello
- *Post-doc*
  John Lazzaro
- *Students*
  Krste Asanović
  David Bailey
  Chris Bregler
  Tim Callahan
  Ben Gomes
  Brian Kingsbury
  Sven Meier
  Srini Narayaran
  Stelios Perissakis
  David Stoutamire
  Su-Lin Wu

Funding

- Office of Naval Research URI Grant (since May 1992)
- National Science Foundation

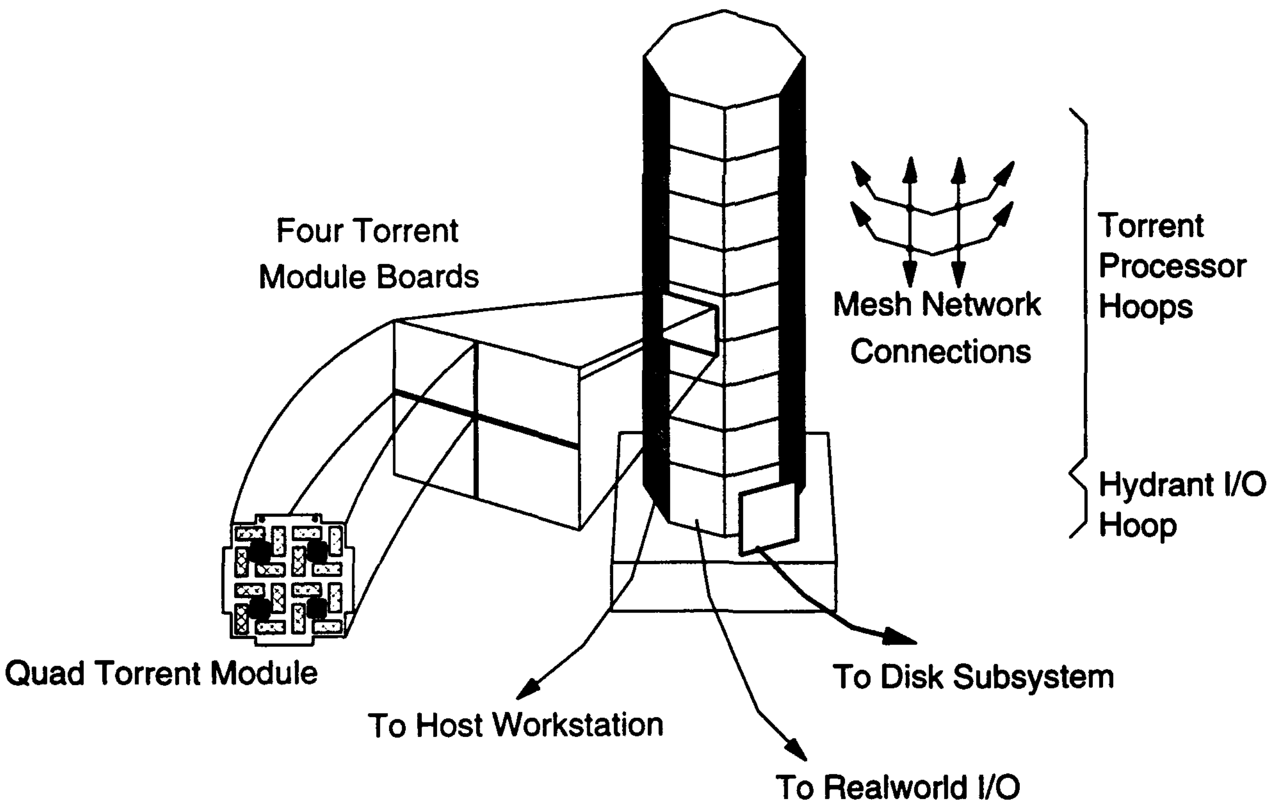
  Experimental Systems
  PYI award
  Graduate Fellowships
  Mammoth Infrastructure Grant
- ICSI

  Funds provided by ministries of research of
  Germany, Italy, and Switzerland, and cooperating companies.
- ARPA/ONR Grant
- Total approximately $2M per year.

Project Overview

Moderately-priced scalable high-performance connectionist computation

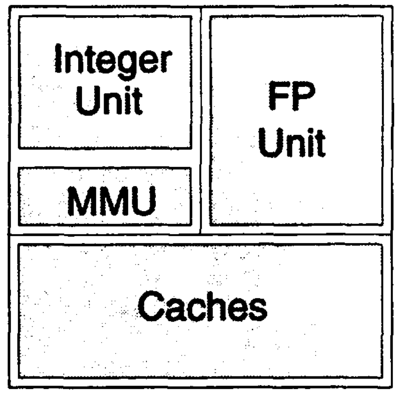
- Tool for Artificial Neural Network Research
  - Training Large Nets (up to 10^9 parameters)
  - Experiments with Real-World I/O (new work in speech and vision)
- Research & Education in Parallel Architectures, VLSI, Connectionist Software.



Benchmark Problem

Evaluate a network with a million units and an average of a thousand connections per unit for a total of a billion connections. This should be done 100 times per second.

Low–Moderate Precision Arithmetic



- Fast Digital Multipliers require $O(N^2)$ area.
- FP makes it worse.
- High-precision arithmetic requires high operand bandwidth.
- Full precision or FP not needed for a wide class of problems
  - Example: NN for speech need only 16b values and 8b weights.

We use 16x16 bit multiplies with 32b accumulates.

Processor Organization

How does one organize many small arithmetic units?

Based on our experience:

1. Amdahl's Law applies
   - Need a well integrated general purpose processor.

   A key issue is finding the right balance of special purpose multiply/add resources and general purpose processing.

2. Hardware should support software and not vice versa.

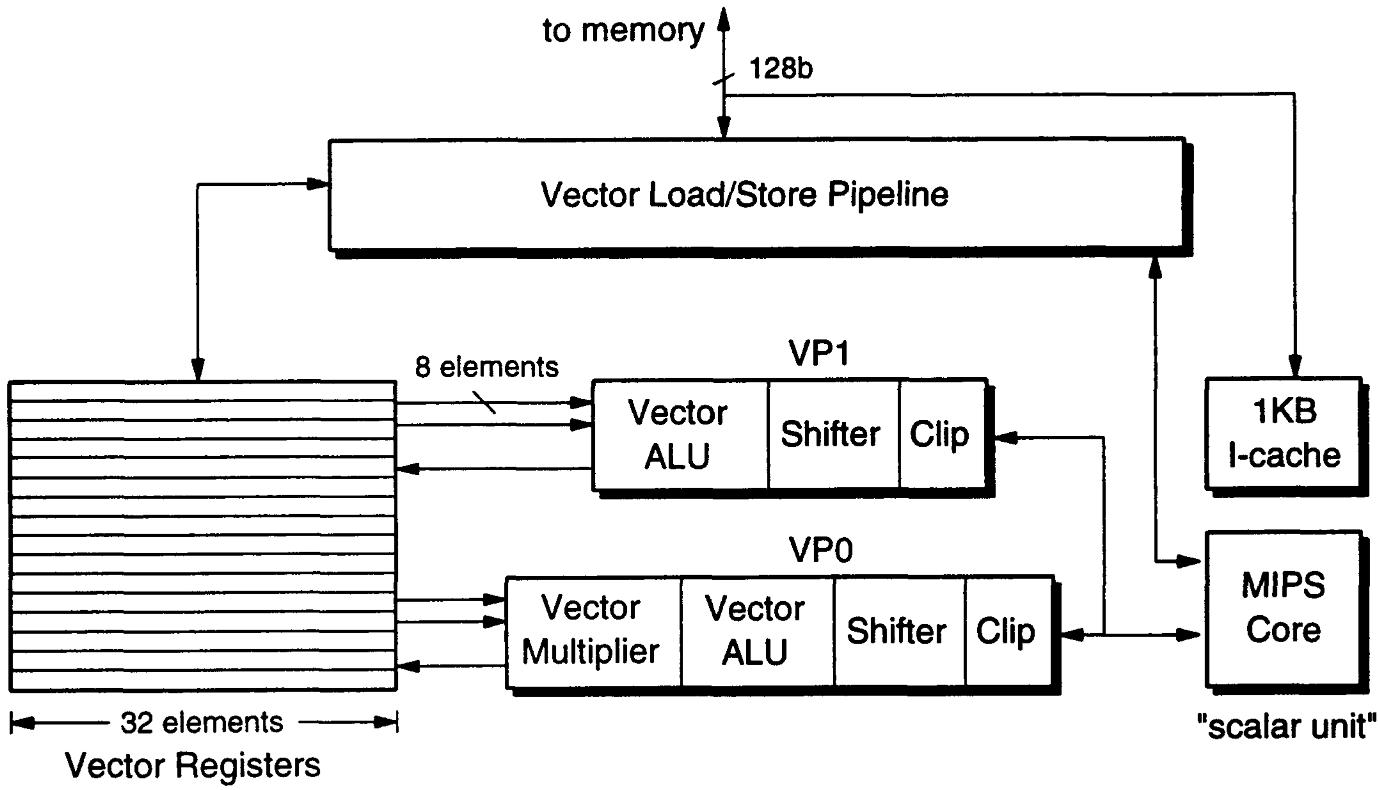
Vector Processing

- **Vector instruction set architecture (ISA)** provides a simple abstraction of highly regular parallelism.
- Many arithmetic operations specified in a single instruction:

[a1 a2 a3 . . . an]	[a1 a2 a3 . . . an]
OP	OP
[b1 b2 b3 . . . bn]	S
↓	↓
[c1 c2 c3 . . . cn]	[c1 c2 c3 . . . cn]

- Well established paradigm
  - Well understood compiler technology
  - Many existing algorithms
- A range of implementation costs and performance are possible (without changing ISA).

T0 Processor



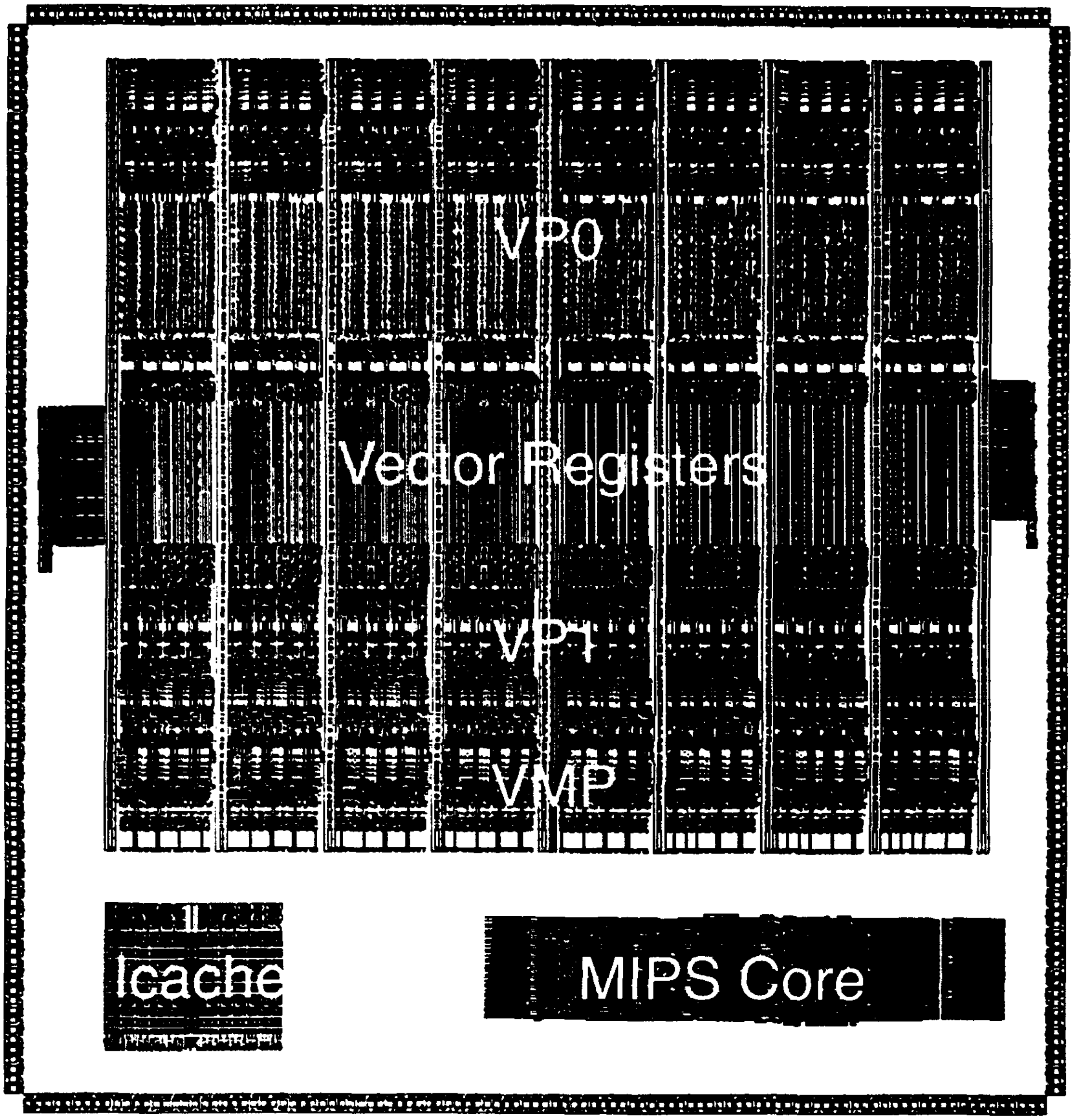
- Vector *load/store* architecture
- Two vector arithmetic datapaths, each 8-way parallel
- Maximum vector length of 32 elements
- Fixed-point and integer (FP emulation)
- Scalar Unit executes MIPS-II instruction set
  - general scalar computations and
  - address generation and control for vector units
- Single instruction issue
- Wide memory bus with various load/store options

T0 System Software

Vector Instructions are implemented as MIPS "coprocessor" instructions $\Rightarrow$

- Standard MIPS programming tools directly usable:
  - C compiler
  - assembler
  - debugger (gdb)
  - system library (standard I/O routines)
- Handcrafted vector libraries for common operations
- Cycle accurate RTL simulator (1100 cycle/sec on SPARC 10/51)
- Fast ISA simulator (100–500K cycles/sec)
- FP emulation for scalar unit
- Yet to come:
  - optimized code scheduler
  - vectorizing compiler
  - FP emulation for vector units
  - Fixed point to FP analyzer

T0 VLSI Implementation

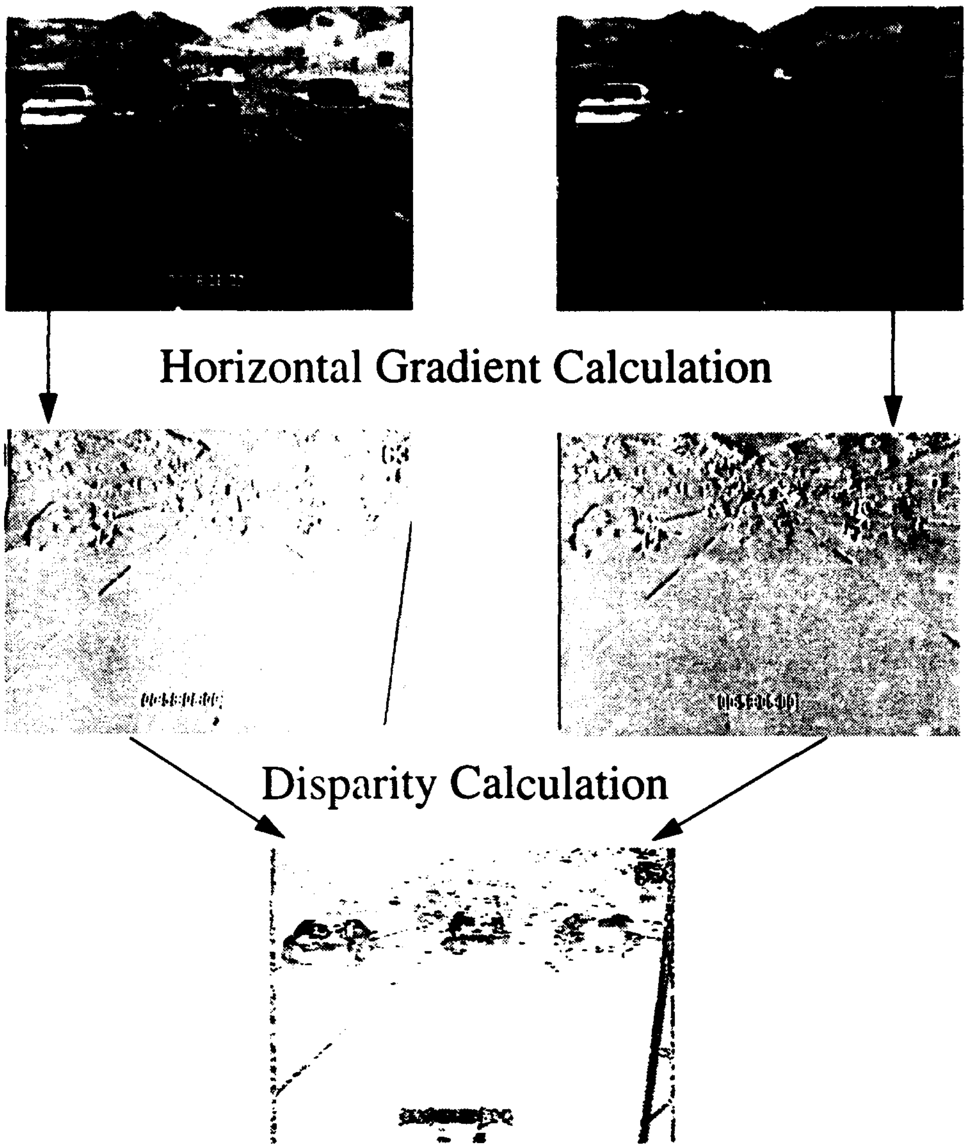


- MOSIS/HP CMOS26B (1.0μm), 50MHz
- 800K transistors
- 800M arithmetic operations per second (vector units only)
- 400M operands/s (800MB/sec) memory bandwidth

T0 Performance

- Neural Network Computations (16b weights, 8b activations):
  - 360 MCPS forward pass
  - 100 MCUPS backprop training
- MPEG decoding (160 X 128 frame):
  - iDCT: 1.02ms
  - (sparc2 implementation: 21.07ms)

T0 for Vision

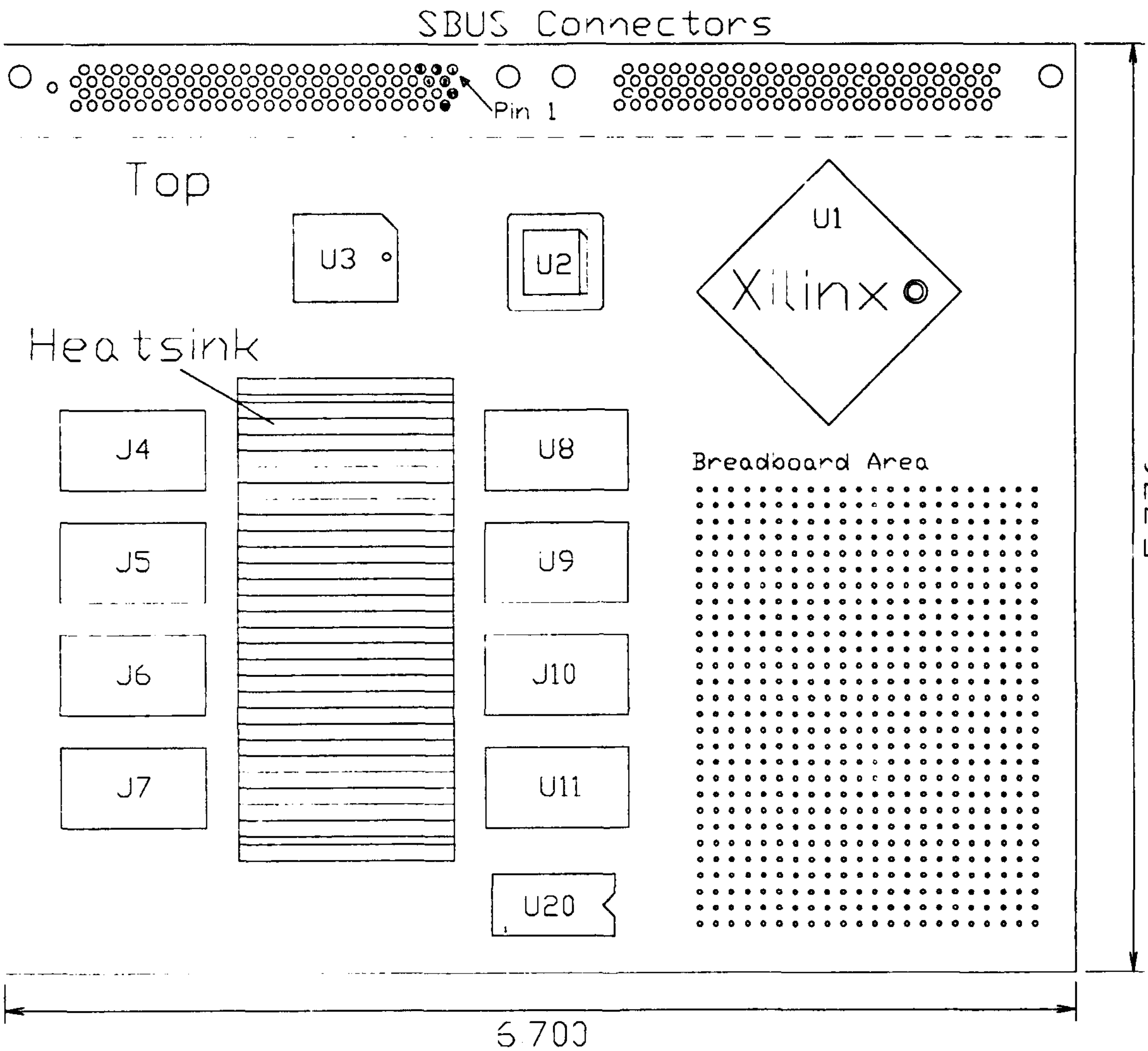


Sample Performance

Task*	Sparc-10 (C++)	T0	T0 Resource Utilization	T0 Performance
Convolution	2.6 sec (5x7 kernel)	0.0172 sec (8x8 kernel)	VP0: .80 VP1: .46 VMP: .56	arith: 500 MOPS/sec mem: 450 MB/sec
Disparity	10.08 sec ("focused")	0.1671 sec ("raw") 0.0668 sec ("focused")	VP0: .20 VP1: .36 VMP: .83	arith: 224 MOPS/sec mem: 664 MB/sec

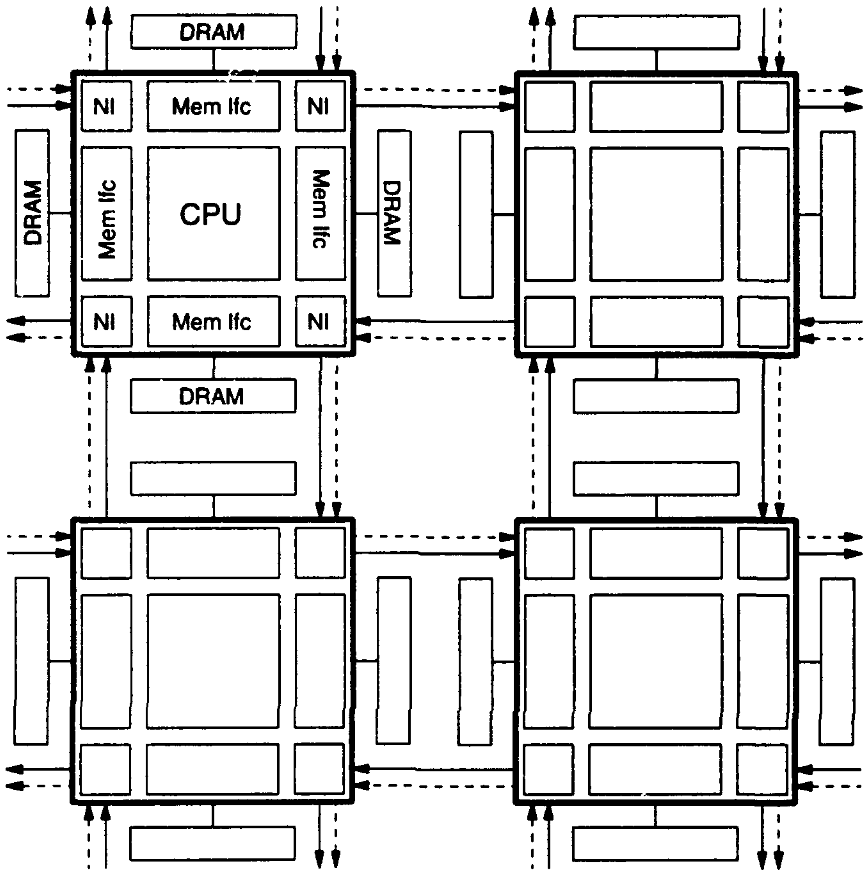
*. on 640 x 480 images

Spert Board



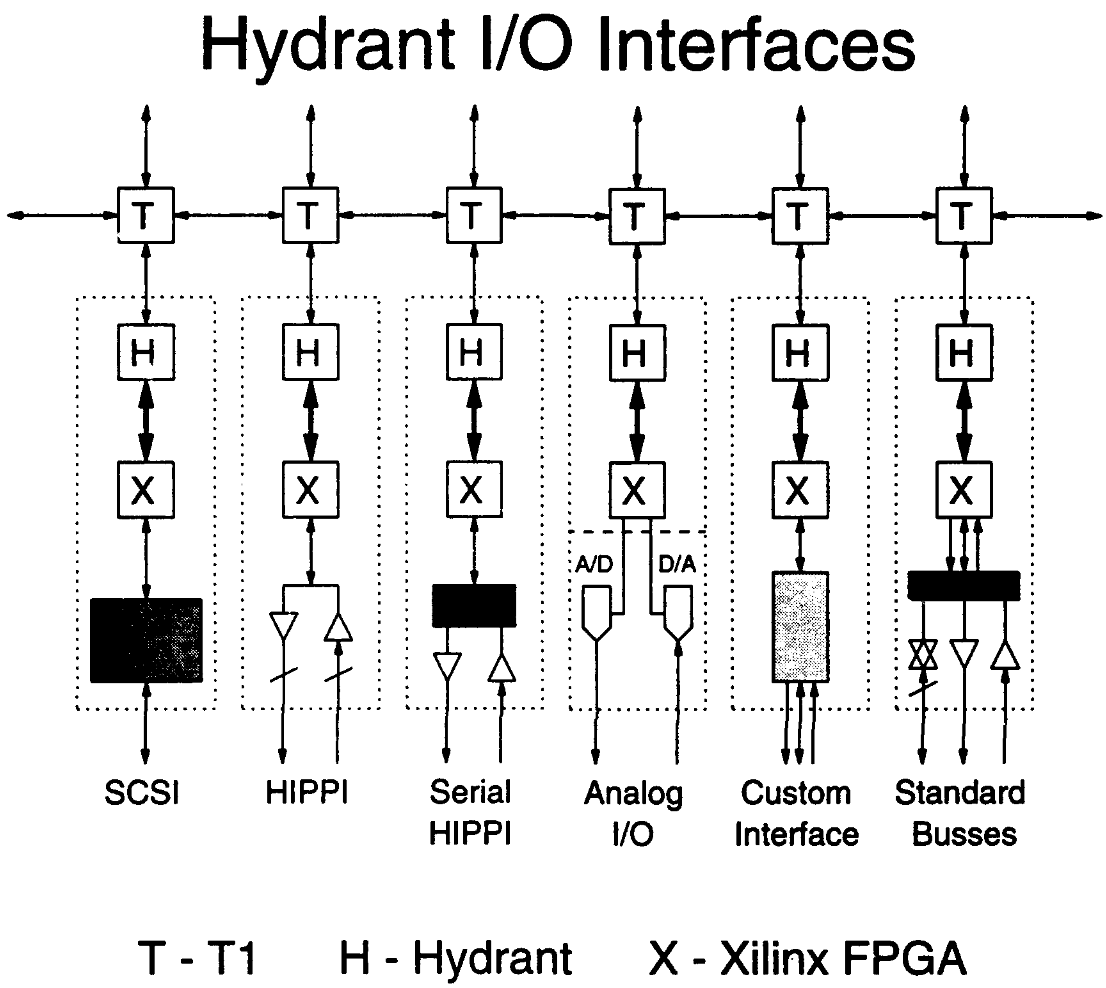
- Sbus (SUN workstation) board
- 33 MBytes/s input / 40 MBytes/s output to Xilinx (DMA)
- 8 MBytes/s I/O via Sbus (programmed I/O) (measured)
- 20MB/s I/O via Sbus DMA (estimate)
- 8 Mbytes fast local SRAM

Multiple Node Systems



- T1 — Node in MIMD massively parallel processor
  - network interface for 2-D mesh connections
  - support for efficient message passing
  - scalable from 1 to 1K nodes

Hydrant I/O Interface



- T1 side — 8b link at 250 Mbytes/s
- Xilinx side —
  - independent address/data pins for each direction
  - 32b wide at 250 Mbytes/s
- RTL level design complete, partial VLSI layout.